MULTIPATH GHOST ELIMINATING EQUALIZER
WITH OPTIMUM NOISE ENHANCEMENT

Technical Field of the Invention

The present invention is directed to an equalizer

5 that substantially eliminates signal ghosts of up to and

including 100% ghosts and, more particularly, to a multipath

equalizer with optimum noise enhancement.

Background of the Invention

Ghosts are produced at a receiver usually because

a signal arrives at the receiver through different

10 transmission paths. For example, in a system having a

single transmitter, the multipath transmission of a signal

may occur because of signal reflection. That is, the

receiver receives a transmitted signal and one or more

reflections (i.e., ghosts) of the transmitted signal. As

15 another example, the multipath transmission of a signal may

occur in a system having plural transmitters that transmit

the same signal to a receiver using the same carrier

frequency. A network which supports this type of

20 transmission is typically referred to as a single frequency

network. Because a ghost results from the multipath

transmission of a signal, a ghost is often referred to as multipath interference.

A variety of systems have been devised to deal with the problems caused by ghosts. For example, spread spectrum systems deal very adequately with the problem of a 100% ghost by spreading the transmitted data over a substantial bandwidth. Accordingly, even though a 100% ghost means that some information may be lost, the data can still be recovered because of the high probability that it was spread over frequencies that were not lost because of the ghost. Unfortunately, the data rate associated with spread spectrum systems is typically too low for many applications.

It is also known to transmit data as a data vector. A matched filter in a receiver correlates the received data with reference vectors corresponding to the possible data vectors that can be transmitted. A match indicates the transmitted data. Unfortunately, the data rate typically associated with the use of matched filters is still too low for many applications.

When high data rates are required, equalizers are often used in a receiver in order to reduce ghosts of a main

signal. An example of a time domain equalizer is a FIR

filter. A FIR filter convolves its response with a received

signal in order to recover data and eliminate any ghosts of

the data. The coefficients applied by the FIR filter

5   asymptotically decrease toward zero for ghosts that are less

than 100%. However, for 100% ghosts, the coefficients

applied by the FIR filter do not asymptotically decrease

toward zero so that a FIR filter becomes infinitely long if

a 100% ghost is to be eliminated, making the FIR filter

10   impractical to eliminate a 100% ghost.

A frequency domain equalizer typically includes a

Fast Fourier Transform (FFT) which is applied to the

received signal. A multiplier multiplies the frequency

domain output of the FFT by a compensation vector. An

15   inverse FFT is applied to the multiplication results in

order to transform the multiplication results to the time

domain. The compensation vector is arranged to cancel the

ghost in the received signal leaving only the main signal.

However, information in the received main signal is lost at

20   certain frequencies so that the output of the inverse FFT

becomes only an approximation of the transmitted data.

U.S. Application 09/158,730 filed September 22,
1998 discloses a vector domain equalizer which effectively
eliminates ghosts up to 100% by distributing the transmitted
data in both time and frequency so that the vectors are
5   essentially random in the time and frequency domains.
Accordingly, in a heavily ghosted channel, all data can be
recovered with small noise enhancement, and any enhanced
noise that does exist is near white. However, the number of
calculations performed by the transform in the receiver to
10   recover the data is large.

U.S. Application Serial No. 09/283,877 filed April
1, 1999 discloses a single path equalizer which effectively
eliminates ghosts up to 100% and which uses fewer
calculations. This equalizer includes a pre-processor, a
15   finite filter, and a post-processor. The pre-processor
multiplies a data input block received from the channel by
coefficients **b** in order to modulate the received main signal
and its ghost so that the ghost is less than the received
main signal. The finite filter applies coefficients **a** in
20   order to eliminate the ghost from the multiplication
results. The post-processor applies coefficients **c** to the
output of the finite filter in order to reverse the effects

of the modulation imposed by the pre-processor. Also, the post-processor applies a window function to the output of the finite filter. This single path equalizer somewhat enhances noise picked up in the channel.

5   U.S. Application Serial No. 09/425,522 filed October 22, 1999 discloses a dual parallel path equalizer having a pre-processor, a finite filter, and a post-processor in each path. The pre-processors multiply a data input block received from the channel by corresponding

10   coefficients b_1 and b_2 in order to modulate the received main signal and its ghost so that the ghost is less than the received main signal. The finite filters apply corresponding coefficients a_1 and a_2 in order to eliminate the ghost from the multiplication results. The post-

15   processors apply corresponding coefficients c_1 and c_2 to the outputs of the finite filters in order to reverse the effects of the modulations imposed by the pre-processors. Each of the outputs of the post-processors is a solution to the problem of a ghost. That is, substantially no ghost is

20   present in the output from each of the post-processors. The outputs of the post-processors are added in order to substantially minimize enhancement of noise, thus producing

better signal to noise performance as compared to a single path equalizer.

The present invention is directed to an equalizer which uses less hardware than the dual path equalizer describe above, and which also substantially eliminates ghosts up to 100% while at the same time producing good noise performance.

Summary of the Invention

In accordance with one aspect of the present invention, a method of equalizing a signal comprises at least the following steps: a) shifting data in a series of input data blocks left by one; b) complex multiplying each of the left shifted data blocks by a first set of equalizer coefficients to provide first adjusted output data blocks, wherein step b) is not a full solution to ghosts; c) complex multiplying each of the input data blocks by a second set of equalizer coefficients to provide second adjusted output data blocks, wherein step c) is not a full solution to ghosts; d) shifting the data in each of the input data blocks of data right by one; e) complex multiplying each of the right shifted input data blocks by a

third set of equalizer coefficients to provide third

adjusted output data blocks, wherein step e) is not a full

solution to ghosts; and, f) adding corresponding ones of

the first, second, and third adjusted output data blocks so

5   as to provide a substantially full solution to ghosts.

In accordance with another aspect of the present

invention, a method is provided to substantially eliminate a

ghost of a received main signal and reduce noise

enhancement. The method comprises the following steps: a)

10 processing the received main signal and the ghost along n

paths to produce n processed main signals and n processed

ghosts, wherein processing along each of the n paths does

not substantially eliminate the ghost, wherein $n > 3$, and

wherein processing along at least some of the n paths

15 includes shifting data; and, b) adding the n processed main

signals and the n processed ghosts such that, because of the

addition of the n processed main signals and the n processed

ghosts, the ghost of the received main signal is

substantially eliminated.

20 In accordance with yet another aspect of the

present invention, an equalizer for processing blocks of

data comprises n processing paths, n - 1 data shifters, n

finite filters, and an adder. The n processing paths are
arranged to process the blocks of data. Each of the n - 1
data shifters is in a corresponding one of the n processing
paths so that one of the n processing path has no data

5   shifter. Each of the n finite filters is in a corresponding
one of the n processing paths, and each of the n finite
filters applies a corresponding set of finite filter
coefficients to the blocks of data. Ghosts of the blocks of
data are not eliminated as a result of the application of

10   the sets of finite filter coefficients corresponding to the
n finite filters, and n > 2. The adder is arranged to add
outputs from the n processing paths. This addition
eliminates ghosts of the blocks of data.

Brief Description of the Drawings

15   These and other features and advantages of the
present invention will become more apparent from a detailed
consideration of the invention when taken in conjunction
with the drawings in which:

Figure 1 illustrates a first embodiment of an

20   equalizer synchronization and coefficient control according
to the present invention;

Figure 2 illustrates a second embodiment of an equalizer synchronization and coefficient control according to the present invention;

Figure 3 illustrates a first embodiment of an equalizer in the form of a dual path equalizer according to the present invention;

Figure 4 illustrates an exemplary set of responses for corresponding elements of Figure 3;

Figure 5 illustrates a second embodiment of an equalizer in the form a dual path equalizer according to the present invention;

Figure 6 illustrates an exemplary set of responses for corresponding elements of Figure 5;

Figure 7 illustrates a third embodiment of an equalizer in the form a dual path equalizer according to the present invention;

Figure 8 illustrates a fourth embodiment of an equalizer in the form a triple path equalizer according to the present invention;

Figure 9 illustrates a fifth embodiment of an equalizer in the form a triple path equalizer according to the present invention;

Figure 10 illustrates a flow graph for the triple path equalizer shown in Figure 9;

Figure 11 illustrates an exemplary set of responses for corresponding elements of Figure 9;

5   Figure 12 illustrates a sixth embodiment of an equalizer in the form a generalized multipath equalizer according to the present invention;

Figure 13 illustrates a seventh embodiment of an equalizer in the form a generalized multipath equalizer 10 according to the present invention;

Figure 14 shows a first embodiment of a converger arranged to control convergence of the dual path equalizer shown in Figure 3;

Figure 15 shows a second embodiment of a converger 15 arranged to control convergence of the dual path equalizer shown in Figure 5;

Figure 16 shows a fourth embodiment of a converger arranged to control convergence of the triple path equalizer shown in Figure 8;

20   Figure 17 shows a fifth embodiment of a converger arranged to control convergence of the triple path equalizer shown in Figure 9;   and,

Figure 18 shows an exemplary error generator that may be used for the error generators of the convergence controls shown in Figures 14, 15, 16, and 17.

Detailed Description

5   SYSTEM DIAGRAMS

As discussed below, the equalizers of the present inventions apply **b**, **a**, and **c** coefficients to received data blocks in order to substantially eliminate ghosts of the data blocks that may be received by a receiver. The **c** coefficients are fixed in both magnitude and width and, thus, require only synchronization with the data blocks. The **b** coefficients are varied in width as the time interval between data blocks and ghosts vary. Thus, the **b** coefficients require both synchronization and width control. The **a** coefficients are varied in width, magnitude, and phase according to both the time interval between data blocks and ghosts and according to the characteristics of the ghosts. Accordingly, the **a** coefficients require synchronization, and width, magnitude, and phase control.

A first embodiment of a synchronization and coefficient control is shown in Figure 1. As shown in

Figure 1, an equalizer 1, such as any of the equalizers

described below, receives data and ghosts of the data. The

data, for example, may be transmitted in the form of data

blocks. The equalizer 1 processes the data blocks and

5   ghosts in order to eliminate ghosts. A synchronizer 2

synchronizes the equalizer 1 to the incoming data blocks

using any conventional synchronizing technique. A channel

estimator 3 estimates the characteristics of the channel

through which the data blocks are transmitted in order to

10   control the **b** coefficients in width and to control the **a**

coefficients in width, magnitude, and phase so as to

substantially eliminate ghosts.

The channel estimator 3, for example, may be any

of the channel estimators that are conventionally used in

15   COFDM systems. Such channel estimators may be used to

estimate the magnitude of ghosts, and the time interval

between data blocks and their ghosts. The magnitude and

time interval can be used as addresses into look up tables

in order to read out the appropriate sets of **a** coefficients.

20   In order to reduce the size of such look up tables to a

manageable number of entries, interpolation may be used

between entries when a combination of the time interval d

and ghost magnitude does not correspond exactly to the addresses of the look up tables. The channel estimator 3 also uses the time interval d to control the width of the **b** coefficients as described below.

5   A second embodiment of a synchronization and coefficient control is shown in Figure 2. As shown in Figure 2, an equalizer 5, such as any of the equalizers described below, receives data and ghosts. The data, for example, may be in the form of data blocks. The equalizer 5 processes the data blocks and ghosts in order to eliminate the ghosts. A synchronizer 6 synchronizes the equalizer 5 to the incoming data blocks using any conventional synchronizing technique. A converger 7 compares the input and output of the equalizer 5 and adjusts the **a** coefficients based upon the comparison results in a direction to substantially eliminate ghosts. The converger 7 also determines the time interval d between data blocks and ghosts in order to control the width of the **b** coefficients. The converger 7 is described in more detail below.

- 13 -

EQUALIZERS

A first equalizer embodiment, i.e., a dual path equalizer 10, is shown in Figure 3 and includes a pre-processor 12 and a 2xFFT 14 (i.e., a twice-the-data-block-length FFT 14) in a common leg of the dual path equalizer 10. The dual path equalizer 10 also includes a first finite filter 16, a first $2xFFT^{-1}$ 18, and a first post-processor 20 in a first path 22, and a second finite filter 24, a second $2xFFT^{-1}$ 26, and a second post-processor 28 in a second path 30. The outputs of the first and second post-processors 20 and 28 are added by an adder 32.

The pre-processor 12 of the dual path equalizer 10 multiplies the signal received from the channel by coefficients b_0. The data may be transmitted in blocks with a guard interval between each adjacent pair of data blocks. The signal processed by the dual path equalizer 10, therefore, includes the data blocks and any ghosts of the data blocks. The coefficients b_0 are arranged as a window function that is substantially coextensive with a received data block and its ghost. Thus, the pre-processor 12 eliminates any energy, primarily noise, that is outside of

the data blocks and their ghosts. The signal received from the channel is designated in Figure 3 as Input Data.

The 2xFFT 14 applies a Fast Fourier Transform to the output of the pre-processor 12. The Fast Fourier

5   Transform has sufficient coefficients so that it is longer than a data block and may be up to twice as long as a data block.

The first finite filter 16 applies coefficients A_1 to the output of the 2xFFT 14. The first finite filter 16

10   may be implemented as a complex multiplier that complex multiplies the coefficients A_1 by the frequency domain output of the 2xFFT 14. (Upper case letters are used to denote the frequency domain, and lower case letters are used to denote the time domain.) The output of the first finite

15   filter 16 includes the data, a modified ghost of the data, and enhanced noise. Unlike the finite filters of the dual path equalizer disclosed in the aforementioned U.S. Application Serial No. 09/425,522, the first finite filter 16 does not eliminate the ghost from the frequency domain

20   output of the pre-processor 12.

The first 2xFFT^{-1} 18 applies an Inverse Fast Fourier Transform to the output of the first finite filter

16.   The Inverse Fast Fourier Transform has sufficient

coefficients so that it is longer than a data block and may

be up to twice as long as a data block.

5   The first post-processor 20 multiplies the time

domain output from the first $2xFFT^{-1}$ 18 by coefficients c_1.

The first post-processor 20 performs a window function to

eliminate any energy, primarily noise, outside of the data

blocks.   This window function has a duration (i.e., width)

which is substantially equal to the duration (i.e., width)

10   of a data block.   Also, the coefficients c_1 applied by the

first post-processor 20 are chosen so as to substantially

minimize noise enhancement.   Unlike the output of the first

post-processor of the dual path equalizer disclosed in the

aforementioned U.S. Application Serial No. 09/425,522, the

output of the first post-processor 20 does not, by itself,

15   represent a solution to the problem of a ghost.

The second finite filter 24 applies coefficients

A_2 to the output of the 2xFFT 14.   The output of the second

finite filter 24 includes the data of the data block, a

20   modified ghost of the data block, and noise.   As in the case

of the first finite filter 16, the second finite filter 24

may be implemented as a complex multiplier and does not

eliminate the ghost from the frequency domain output of the 2xFFT 14.

The second 2xFFT^{-1} 26 applies an Inverse Fast Fourier Transform to the output of the second finite filter 24. The Inverse Fast Fourier Transform has sufficient coefficients so that it is longer than a data block and may be up to twice as long as a data block.

The second post-processor 28 multiplies the time domain output from the second 2xFFT^{-1} 26 by coefficients c_2. The second post-processor 28 performs a window function to eliminate any energy in the received signal that is outside of the data blocks. This window function has a duration which is substantially equal to the duration of a data block. Also, the coefficients c_2 applied by the second post-processor 28 are chosen so as to substantially minimize noise enhancement. As in the case of the first post-processor 20, the output of the second post-processor 28 does not represent a solution to the problem of a ghost.

As discussed above, the outputs of the first and second post-processors 20 and 28 are added by the adder 32. As a result, the data blocks emerging from each of the first and second paths 22 and 30 are correlated and add to produce

a larger amplitude. The ghosts in the outputs of the first and second post-processors 20 and 28 substantially cancel because of the application of the coefficients A_1, A_2, c_1, and c_2. Noise (such as white noise) is less correlated than the data blocks so that, when the outputs of the first and second post-processors 20 and 28 are added, the noise adds to a less extent than does the data blocks. Thus, noise enhancement is substantially minimized.

Thus, although neither of the outputs of the first and second post-processors 20 and 28 is a solution to channel distortion such as ghosts up to 100%, the combined output from the adder 32 is a solution, so that the dual path equalizer 10 adequately deals with ghosts up to and including a 100% ghost. Additionally, the signal to noise ratio of the dual path equalizer 10 is improved over a single path equalizer.

Exemplary sets of the coefficients b_0, A_1, A_2, c_1, and c_2 are shown in Figure 4. The first column of Figure 4 shows the real parts of the coefficients b_0, A_1, A_2, c_1, and c_2, the second column of Figure 4 shows the imaginary parts of the coefficients b_0, A_1, A_2, c_1, and c_2, and the third

column of Figure 4 shows the absolute values of the coefficients b_0, A_1, A_2, c_1, and c_2.

As can be seen from Figure 4, and as discussed above, the coefficients b_0 applied by the pre-processor 12 have a unity real part and a zero imaginary part that are arranged as a window function having a duration that is substantially coextensive with a received data block and its ghost. The number of coefficients in the set of coefficients b_0 depends upon the size of the data block and the interval d between the signal and its ghost. As shown in Figure 4, for example, if there are sixteen samples in a data block and the interval d between the data block and its ghost is 1 sample (i.e., 1/16 of a data block), then there are seventeen coefficients (16 + 1) in the set of coefficients b_0. The width of (i.e., the number of coefficients in) the set of coefficients b_0, therefore, is varied as the interval d between the data block and its ghost varies.

The coefficients c_1 and c_2 applied respectively by the first and second post-processors 20 and 28 have oppositely sloped real parts and zero imaginary parts that are arranged as corresponding window functions. The

coefficients c_1 and c_2 are sloped, linear, and reversed

weighting functions that operate to optimize noise in such a

way that the signal to noise ratio at the output of the

adder 32 is substantially enhanced. The width of the

5   coefficients c_1 is fixed and is equal to the width of a data

block. Thus, if a data block has sixteen samples, then

there are sixteen coefficients c_1. Similarly, the width of

the coefficients c_2 is fixed and is equal to the width of a

data block. Thus, if a data block has sixteen samples, then

10   there are sixteen coefficients c_2.

The coefficients A_1 and A_2 applied respectively by

the first and second finite filters 16 and 24 have non-zero

real and imaginary parts. As discussed below, the

coefficients A_1 and A_2 are adjusted during operation of the

15   dual path equalizer 10 so that, when the outputs of the

first and second paths 22 and 30 are added by the adder 32,

ghosts are substantially eliminated and noise enhancement is

substantially minimized. Each of the coefficients A_1 and A_2

has a width that is up to twice as long as the width of a

20   data block. Thus, if the data block has sixteen samples,

for example, there may be up to thirty-two coefficients for

each of the coefficients A_1 and A_2.

- 20 -

The pre-processor 12 eliminates noise that may
exist outside of the data block and its ghost. Thus, the
pre-processor 12 could be eliminated from the dual path
equalizer 10. However, if so, noise in the output of the
dual path equalizer 10 will be greater than if the pre-
processor 12 were not eliminated.

The dual path equalizer 10 includes only one pre-
processor and only one FFT. Accordingly, there is less
hardware in the dual path equalizer 10 than in the dual path
equalizer disclosed in the aforementioned application
09/425,522.

As a second equalizer embodiment, a dual path
equalizer 40 is shown in Figure 5 and includes a pre-
processor 42 and a 2xFFT 44 in a common leg. The dual path
equalizer 40 also includes a first finite filter 46 and a
first post-processor 48 in a first path 50, and a second
finite filter 52 and a second post-processor 54 in a second
path 56. The outputs of the first and second post-
processors 48 and 54 are added by an adder 58, and the
output of the adder 58 is down sampled by two by a down
sampler 59. Down sampling is needed to reduce the over
sampling data produced by the 2xFFT 44 in a manner

equivalent to that effected by the window functions applied

by the first and second post-processors 20 and 28 of Figure

3.

The pre-processor 42 of the dual path equalizer 40

5   multiplies the signal received from the channel by

coefficients b_0. As in the case of the coefficients b_0

applied by the pre-processor 12, the coefficients b_0 applied

by the pre-processor 42 are arranged as a window function

that is substantially coextensive with a received data block

10   and its ghost.

The 2xFFT 44 applies a Fast Fourier Transform to

the output of the pre-processor 42. The Fast Fourier

Transform has sufficient coefficients so that it is longer

than a data block and may be up to twice as long as a data

15   block.

The first finite filter 46 applies coefficients A_1

to the output of the 2xFFT 44. The first finite filter 46

may be implemented as a complex multiplier that complex

multiplies the coefficients A_1 by the frequency domain

20   output of the 2xFFT 44. As in the case of the first and

second finite filters 16 and 24 of Figure 3, the first

finite filter 46 does not eliminate ghosts from the frequency domain output of the 2xFFT 44.

The first post-processor 48 convolves the frequency domain output from the first finite filter 46 with

5   coefficients C_1. As in the case of the first and second post-processors 20 and 28 of Figure 3, the output of the first post-processor 48 does not represent a solution to the problem of a ghost. However, the first post-processor 48 weights the output of the first finite filter 46 in order to

10 substantially optimize noise in the first path 50. Accordingly, when the output of the first post-processor 48 is combined with the output of the second post-processor 54 by the adder 58, ghosts are substantially eliminated in the output of the adder 58, and noise enhancement is

15 substantially minimized.

The second finite filter 52 applies coefficients A_2 to the output of the 2xFFT 44. The second finite filter 52 may be implemented as a complex multiplier that complex multiplies the coefficients A_2 by the frequency domain

20 output of the 2xFFT 44. As in the case of the first finite filter 46, the second finite filter 52 does not eliminate

ghosts from the frequency domain output of the pre-processor
42.

The second post-processor 54 convolves the
frequency domain output from the second finite filter 52

5   with coefficients C_2. As in the case of the first post-
processor 48, the output of the second post-processor 54
does not represent a solution to the problem of a ghost.
However, the second post-processor 54 weights the output of
the second finite filter 52 in order to substantially

10   optimize the noise picked up from the channel by the
received signal. Accordingly, as discussed above, when the
output of the first post-processor 48 is combined with the
output of the second post-processor 54 by the adder 58,
ghosts are substantially eliminated in the output of the

15   adder 58, and noise enhancement is substantially minimized.

As indicated above, the outputs of the first and
second post-processors 48 and 54 are added by the adder 58.
As a result, the data blocks emerging from each of the first
and second paths 50 and 56 are correlated and add to produce

20   a larger amplitude. The ghosts in the outputs of the first
and second post-processors 48 and 54 substantially cancel
because of the application of the coefficients A_1, A_2, C_1,

- 24 -

and c_2. Noise (such as white noise) is less correlated than the data blocks so that, when the outputs of the first and second post-processors 48 and 54 are added, the noise adds to a less extent than does the data blocks. Thus, noise

5   enhancement is substantially minimized.

Accordingly, although neither of the outputs of the first and second post-processors 48 and 54 is a solution to channel distortion such as ghosts up to the 100%, the combined output from the adder 58 is a solution, so that the

10   dual path equalizer 40 adequately deals with ghosts up to and including a 100% ghost. Additionally, the signal to noise ratio of the dual path equalizer 40 is improved over a single path equalizer.

The dual path equalizer 40 does not include the

15   first $2xFFT^{-1}$ 18 and the second $2xFFT^{-1}$ 26 of the dual path equalizer 10. Accordingly, there is less hardware in the dual path equalizer 40 than in the dual path equalizer 10. Also, because there is a Fast Fourier Transform but no Inverse Fast Fourier Transform upstream of the first and

20   second post-processors 48 and 54, the first and second post-processors 48 and 54 are arranged to operate in the frequency domain. Because the dual path equalizer 40 has an

FFT but no inverse FFTs, an inverse FFT should be included in the transmitter that transmits the data blocks to the dual path equalizer 40.

Exemplary sets of the coefficients b_0, A_1, A_2, C_1, and C_2 are shown in Figure 6 for an interval or delay of 1/16 between a data block and its ghost. The first column of Figure 6 shows the real parts of the coefficients b_0, A_1, A_2, C_1, and C_2, the second column of Figure 6 shows the imaginary parts of the coefficients b_0, A_1, A_2, C_1, and C_2, and the third column of Figure 6 shows the absolute values of the coefficients b_0, A_1, A_2, C_1, and C_2. The coefficients b_0 applied by the pre-processor 42 may be the same as the coefficients b_0 applied by the pre-processor 12. As can be seen from Figure 6, and as discussed above, the coefficients b_0 applied by the pre-processor 42 have a unity real part and a zero imaginary part that are arranged as a window function having a duration that is substantially coextensive with a received data block and its ghost.

The coefficients A_1 and A_2 applied respectively by the first and second finite filters 46 and 52 have non-zero real and imaginary parts. As discussed below, the coefficients A_1 and A_2 are adjusted during operation of the

dual path equalizer 40 so that, when the outputs of the

first and second paths 50 and 56 are added by the adder 58,

ghosts are substantially eliminated and noise enhancement is

substantially minimized.

5   The coefficients C_1 applied by the first post-

processor 48 may be derived by performing a Fast Fourier

Transform of the coefficients c_1 applied by the first post-

processor 20 of the dual path equalizer 10. A predetermined

number of the resulting coefficients may then be used as the

10   coefficients C_1. For example, as shown in Figure 6, the

resulting seven most significant coefficients are used as

the coefficients C_1. However, more of the coefficients

resulting from the Fast Fourier Transform of the

coefficients c_1 applied by the first post-processor 20 could

15   be used in order to more effectively eliminate ghosts and

minimize noise enhancement.

          Similarly, the coefficients C_2 applied by the

second post-processor 56 may be derived by performing a Fast

Fourier Transform of the coefficients c_2 applied by the

20   second post-processor 28 of the dual path equalizer 10. A

predetermined number of the resulting coefficients may then

be used as the coefficients C_2. For example, as shown in

Figure 6, the resulting seven most significant coefficients

are used as the coefficients C_2. However, more of the

coefficients resulting from the Fast Fourier Transform of

the coefficients c_2 applied by the second post-processor 28

5   could be used if desired.

As a third equalizer embodiment, a dual path

equalizer 60 is shown in Figure 7 and includes a pre-

processor 62 and a 2xFFT 64 in a common leg. The dual path

equalizer 60 also includes a first finite filter 66 and a

10   first post-processor 68 in a first path 70, and a second

finite filter 72 and a second post-processor 74 in a second

path 76. The outputs of the first and second post-

processors 68 and 74 are added by an adder 78, and the

output of the adder 78 is down sampled by two by a down

15   sampler 79.

As can be seen by comparing Figures 5 and 7, the

hardware of the dual path equalizer 60 is quite similar to

the hardware of the dual path equalizer 40. However, the

coefficients applied by the pre-processor 62, by the first

20   and second finite filters 66 and 72, and by the first and

second post-processors 68 and 74 are different.

The coefficients b_0 applied by the pre-processor 62 are curved as can be seen in Figure 7. The coefficients b_0 are curved according to the function $1/(2 - \cos(t))$. The coefficients b_0 applied by the pre-processor 62 are curved in order to minimize the complexity of the coefficients C_1 and C_2 applied by the first and second post-processors 48 and 54 of the dual path equalizer 40 shown in Figure 5. This curvature of the coefficients b_0 permits implementation of the first and second post-processors 68 and 74 as simplified two tap convolvers. Each tap of each convolver has a magnitude of one which permits the first and second post-processors 68 and 74 to be implemented as adders rather than as multipliers. As in the case of the prior embodiments, the width of (i.e., the number of coefficients in) the set of coefficients b_0 is varied as the interval d between the data block and its ghost varies. As the width of the set of coefficients b_0 varies, the curvature of the coefficients b_0 still follows the function $1/(2 - \cos(t))$.

The coefficients A_1 and A_2 applied respectively by the first and second finite filters 66 and 72 have non-zero real and imaginary parts. As discussed below, the coefficients A_1 and A_2 are adjusted during operation of the

dual path equalizer 60 so that, when the outputs of the first and second paths 70 and 76 are added by the adder 78, ghosts are substantially eliminated and noise enhancement is substantially minimized.

5   The coefficients C_1 applied by the first post-processor 68 are such that application of an Inverse Fast Fourier Transform to the coefficients C_1 results in a set of coefficients in the time domain that have a curvature corresponding to the curvature of the coefficients b_0. Similarly, the coefficients C_2 applied by the second post-processor 74 are such that application of an Inverse Fast Fourier Transform to the coefficients C_2 results in a set of coefficients in the time domain that have a curvature corresponding to the curvature of the coefficients b_0.

        It may be noted that the right most tap of the coefficients C_1 and the left most tap of the coefficients C_2 coincide. Accordingly, the coefficients C_1 and C_2 may be de-composed into three sets of coefficients and may be applied by three corresponding post-processors as shown in 20   Figure 8. Figure 8 shows a fourth equalizer embodiment in the form of a triple path equalizer 80 which includes a pre-processor 82 and a 2xFFT 84 in a common leg. The triple

path equalizer 80 also includes a first finite filter 86 and a first post-processor 88 in a first path 90, a second finite filter 92 and a second post-processor 94 in a second path 96, and a third finite filter 98 and a third post-

5 processor 100 in a third path 102. The outputs of the first, second, and third post-processors 88, 94, and 100 are added by an adder 104. A down sampler 106 down samples by two the output of the adder 104.

The coefficients b_0 applied by the pre-processor

10 82 are the same as the coefficients b_0 applied by the pre-processor 62 of the dual path equalizer 60. The coefficients A_1 applied by the first finite filter 86, the coefficients A_2 applied by the second finite filter 92, and the coefficients A_3 applied by the third finite filter 98 are adjusted during operation of the triple path equalizer

15 80 so that, when the outputs of the first, second, and third paths 90, 96, and 102 are added by the adder 104, ghosts are substantially eliminated and noise enhancement is substantially minimized. The coefficient C_1 applied by the

20 first post-processor 88 is the left-hand value of the coefficients C_1 applied by the first post-processor 68, the coefficient C_2 applied by the second post-processor 94 is

the common value of the coefficients C_1 and C_2 applied by the first and second post-processors 68 and 74, and the coefficient C_3 applied by the third post-processor 100 is the right-hand value of the coefficients C_2 applied by the second post-processor 74.

The single tap of the first post-processor 88 results in a shift of one sample to the left of the data processed in the first path 90, the single tap of the second post-processor 94 results in no sample shift of the data processed in the second path 96, and the single tap of the third post-processor 100 results in a shift of one sample to the right of the data processed in the third path 102. Therefore, as shown by a triple path equalizer 110 of Figure 9, the first, second, and third post-processors 88, 94, and 100 can be replaced by sample shifters.

Accordingly, the triple path equalizer 110 includes a pre-processor 112 and a 2xFFT 114 in a common leg. The triple path equalizer 110 also includes a left one-sample shifter 116, a first by-two down sampler 118, and a first finite filter 120 in a first path 122, a second by-two down sampler 124 and a second finite filter 126 in a second path 128, and a right one-sample shifter 130, a third

by-two down sampler 132, and a third finite filter 134 in a third path 136. The outputs of the first, second, and third finite filters 120, 126, and 134 are added by an adder 138.

The coefficients b_0 applied by the pre-processor

5   112 are the same as the coefficients b_0 applied by the pre-processor 82 of the triple path equalizer 80. The coefficients A_1 applied by the first finite filter 120, the coefficients A_2 applied by the second finite filter 126, and the coefficients A_3 applied by the third finite filter 134

10   are adjusted during operation of the triple path equalizer 110 so that, when the outputs of the first, second, and third paths 122, 128, and 136 are added by the adder 138, ghosts are substantially eliminated and noise enhancement is substantially minimized.

     Figure 10 illustrates a flow graph for the triple path equalizer 110 shown in Figure 9. The variable n designates the number of the sample emerging from the 2xFFT 114. To produce an output sample D_n from the adder 138, the first finite filter 120 processes a sample H_{2n-1}, the second

20   finite filter 126 processes a sample H_{2n}, and the third finite filter 134 processes a sample H_{2n+1}. Because of the second by-two down sampler 124, the second finite filter 126

processes only the even samples of the samples supplied by the 2xFFT 114. Because of the left one-sample shifter 116 and the first by-two down sampler 118, the first finite filter 120 processes the sample H_{2n-1} which is one odd sample behind the sample H_{2n}. Because of the right one-sample shifter 130 and the third by-two down sampler 132, the third finite filter 134 processes the sample H_{2n+1} which is one odd sample ahead of the sample H_{2n}.

Exemplary sets of the coefficients b_0, A_1, A_2, and A_3, are shown in Figure 11 assuming an interval d between a data block and its ghost of one sample (i.e., 1/16 of a data block containing sixteen samples). The first column of . Figure 6 shows the real parts of the coefficients b_0, A_1, A_2, and A_3, the second column of Figure 11 shows the imaginary parts of the coefficients b_0, A_1, A_2, and A_3, and the third column of Figure 11 shows the absolute values of the coefficients b_0, A_1, A_2, and A_3.

The coefficients b_0 applied by the pre-processor 112 are the same as the coefficients b_0 applied by the pre-processor 62. As in the case of the triple path equalizer 80, the width of (i.e., the number of coefficients in) the set of coefficients b_0 is varied as the interval d between

- 34 -

the data block and its ghost varies. As the width of the

set of coefficients b_0 varies, the curvature of the

coefficients b_0 still follows the function $1/(2 - \cos(t))$.

As can be seen from Figure 11, and as discussed

5   above, the coefficients b_0 applied by the pre-processor 112

have a real part shaped according to the function $1/(2 -$

$\cos(t))$ and a zero imaginary part and are arranged as a

window function having a duration that is substantially

coextensive with a received data block and its ghost. The

10   coefficients A_1 applied by the first finite filter 120, the

coefficients A_2 applied by the second finite filter 126, and

the coefficients A_3 applied by the third finite filter 134

are adjusted during operation of the triple path equalizer

110 so that, when the outputs of the first, second, and

15   third paths 122, 128, and 136 are added by the adder 138,

ghosts are substantially eliminated and noise enhancement is

substantially minimized.

The triple path equalizer 110 shown in Figure 9

can be generalized by expanding the number of paths.

20   Accordingly, a multipath equalizer 140 shown in Figure 12

includes a pre-processor 142 and a 2xFFT 144 in a common

leg. The multipath equalizer 140 also includes a plurality

of paths ... 146_{n-2}, 146_{n-1}, 146_n, 146_{n+1}, 146_{n+2} ... having

corresponding sample shifters ... 148_{n-2}, 148_{n-1}, 148_{n+1},

148_{n+2} ..., corresponding by-two down samplers ... 150_{n-2},

150_{n-1}, 150_n, 150_{n+1}, 150_{n+2} ..., and corresponding finite

5   filters ... 152_{n-2}, 152_{n-1}, 152_n, 152_{n+1}, 152_{n+2} .... The

outputs of the finite filters ... 152_{n-2}, 152_{n-1}, 152_n,

152_{n+1}, 152_{n+2} ... are added by an adder 154.

The coefficients b_0 applied by the pre-processor

142 are the same as the coefficients b_0 applied by the pre-

10   processor 82 of the triple path equalizer 80. As in the

case of the triple path equalizer 80, the width of (i.e.,

the number of coefficients in) the set of coefficients b_0 is

varied as the interval d between the data block and its

ghost varies. As noted above, the window function applied

15   by the pre-processor 142 eliminates any energy, primarily

noise, that is outside of the data blocks and their ghosts.

The pre-processor 142 can be eliminated. However, if the

pre-processor 142 is eliminated, the finite filters ...

152_{n-2}, 152_{n-1}, 152_n, 152_{n+1}, 152_{n+2} ... will process more

20   noise than would otherwise be the case. Thus, without the

pre-processor 142, additional noise may be present in the

output of the adder 154.

The coefficients ... A_{n-2}, A_{n-1}, A_n, A_{n+1}, A_{n+2} ...
applied by the corresponding finite filters ... 152_{n-2}, 152_{n-1}, 152_n, 152_{n+1}, 152_{n+2} ... are adjusted during operation of
the multipath equalizer 140 so that, when the outputs of the
paths ... 146_{n-2}, 146_{n-1}, 146_n, 146_{n+1}, 146_{n+2} ... are added by
the adder 154, ghosts are substantially eliminated and noise
enhancement is substantially minimized.

Similarly, the dual path equalizer 40 as shown in
Figure 5 can be generalized by expanding the number of
paths. Accordingly, a multipath equalizer 160 shown in
Figure 13 includes a pre-processor 162 and a 2xFFT 164 in a
common leg. The multipath equalizer 160 also includes
finite filters ... 166_{n-2}, 166_{n-1}, 166_n, 166_{n+1}, 166_{n+2} ... and
corresponding post-processors ... 168_{n-2}, 168_{n-1}, 168_n,
168_{n+1}, 168_{n+2} ... arranged in corresponding paths ... 170_{n-2},
170_{n-1}, 170_n, 170_{n+1}, 170_{n+2} .... The outputs of the post-
processors ... 168_{n-2}, 168_{n-1}, 168_n, 168_{n+1}, 168_{n+2} ... are
added by an adder 172, and the output of the adder 172 is
down sampled by two by a down sampler 174.

The finite filters ... 166_{n-2}, 166_{n-1}, 166_n, 166_{n+1},
166_{n+2} ... may be implemented as complex multipliers that
complex multiply the coefficients ... A_{n-2}, A_{n-1}, A_n, A_{n+1},

- 37 -

A_{n+2} ... by the frequency domain output of the 2xFFT 164.

The post-processors ... 168_{n-2}, 168_{n-1}, 168_n, 168_{n+1}, 168_{n+2}

... convolve the frequency domain outputs from the finite

filters ... 166_{n-2}, 166_{n-1}, 166_n, 166_{n+1}, 166_{n+2} ... 46 with

coefficients ... C_{n-2}, C_{n-1}, C_n, C_{n+1}, C_{n+2} ....

The coefficients ... A_{n-2}, A_{n-1}, A_n, A_{n+1}, A_{n+2} ...

applied by the corresponding finite filters ... 166_{n-2}, 166_{n-1}, 166_n, 166_{n+1}, 166_{n+2} ... are adjusted during operation of

the multipath equalizer 160 so that, when the outputs of the

paths ... 170_{n-2}, 170_{n-1}, 170_n, 170_{n+1}, 170_{n+2} ... are added by

the adder 172, ghosts are substantially eliminated and noise

enhancement is substantially minimized.

The coefficients ... C_{n-2}, C_{n-1}, C_n, C_{n+1}, C_{n+2} ...

may include the coefficients C_1 and C_2 applied by the first

and second post-processors 48 and 54 of the dual path

equalizer 40 and additional coefficients selected to achieve

a desired level of performance. Also, the pre-processor 162

can be eliminated. However, if the pre-processor 162 is

eliminated, the finite filters ... 166_{n-2}, 166_{n-1}, 166_n,

166_{n+1}, 166_{n+2} ... will process more noise than would

otherwise be the case. Thus, without the pre-processor 162,

additional noise may be present in the output of the adder 172.

One of the advantages of the multipath equalizers 140 and 160 is that, by increasing the number of finite

5   filters, any weighting functions characterizing the pre-processors 142 and 162 can be transferred to the finite filters. Accordingly, the pre-processors 142 and 162 can be eliminated. Alternatively, the weighting of the coefficients of the pre-processors 142 and 162 can be

10   eliminated so that these coefficients form pure window functions.

CONVERGENCE

As discussed above, the **A** coefficients may be adaptively controlled by a converger 7 to ensure that the actual output of the equalizer converges on the correct output, i.e., an output free of ghosts. A first embodiment of the converger 7, i.e., an adaptive coefficient control 180, may be provided for the dual path equalizer 10 and is

20   shown in Figure 14. The adaptive convergence control 180 includes a conjugater 182 which conjugates the data from the output of the 2xFFT 14 to facilitate the use of an LMS

algorithm to converge the **A** coefficients of the equalizer

10.   The data exiting the 2xFFT 14 is complex data.   The

conjugater 182 conjugates this data by reversing the sign of

the imaginary part of the data.

5   The output of the conjugater 182 is supplied to

first and second correlators 184 and 186.   An error

generator 188, which is discussed more fully below,

generates an error based upon the output data from the dual

path equalizer 10.   This error must be processed in the same

10 manner as the output of the finite filters 16 and 24 of the

dual path equalizer 10.   Therefore, the adaptive coefficient

control 180 includes an inverse first post-processor 190 and

a 2xFFT 192.   Also, the adaptive coefficient control 180

includes an inverse second post-processor 194 and a 2xFFT

15 196.   The inverse first post-processor 190 and the inverse

second post-processor 194 produce the inverse of the first

and second post-processors 20 and 28.   Additionally, the

2xFFT 192 and the 2xFFT 196 produce the inverse of the first

and second $2xFFT^{-1}$ 18 and $2xFFT^{-1}$ 26 of the dual path

20 equalizer 10.   The outputs of the first and second 2xFFTs

192 and 196 are supplied to the corresponding correlators

184 and 186.

The correlators 184 and 186 multiply (i) the error from the error generator 188 as processed by the respective inverse first and second post-processors 190 and 194 and the respective first and second 2xFFTs 192 and 196 and (ii) the conjugated output of the 2xFFT 14. This multiplication effects a modified LMS algorithm to produce adjustments values for the A coefficients to converge the equalizer 10. In other words, the outputs of the correlators 184 and 186 are adjustment values of the coefficients A_1 and A_2 that, when added to the existing coefficients A_1 and A_2 applied by the dual path equalizer 10, would cause the dual path equalizer 10 to eliminate ghosts from the received signal.

However, instead of correcting the coefficients A_1 and A_2 in one operation, the coefficients A_1 and A_2 are adjusted in increments. Therefore, multipliers 198 and 200 multiply the outputs of the correlators 184 and 186 by a quantity α, which has a value of less that one. The output of the multiplier 198 is added to the existing coefficients A_1 of the first finite filter 16, and the output of the multiplier 200 is added to the existing coefficients A_2 of the second finite filter 28. The value α is used so that

these coefficients A_1 and A_2 are corrected in small
increments in order to ensure a smooth convergence.

A second embodiment of the converger 7, i.e., an
adaptive coefficient control 210, may be provided for the

5   dual path equalizer 40 and is shown in Figure 15. The
adaptive convergence control 210 includes a conjugater 212
which conjugates the data from the 2xFFT 44. The output of
the conjugater 212 is supplied to first and second
correlators 214 and 216. An error generator 218, which is

10   shown in exemplary form in Figure 18, generates an error
based upon the output data of the dual path equalizer 40.
This error must be processed in the same manner as the
output of the finite filters 46 and 52 of the dual path
equalizer 40. Therefore, the adaptive coefficient control

15   210 includes an up sampler 220 which up samples by two the
output of the error generator 218 in order to thereby
reverse the effects of the down sampler 59. The adaptive
coefficient control 210 also includes an inverse first post-
processor 222 and an inverse second post-processor 224. The

20   inverse first post-processor 222 and the inverse second
post-processor 224 produce an inverse of the first and
second post-processors 48 and 54. The outputs of the

inverse first and second post-processors 222 and 224 are supplied to the corresponding correlators 214 and 216.

The correlators 214 and 216 multiply (i) the up sampled error from the error generator 218 as processed by the respective inverse first and second post-processors 222 and 224 and (ii) the conjugated reference data. In effect, the outputs of the correlators 214 and 216 are adjustment values of the coefficients A_1 and A_2 that, when added to the existing coefficients A_1 and A_2 applied by the dual path equalizer 40, would cause the dual path equalizer 40 to eliminate ghosts from the received signal.

Again, instead of correcting the coefficients A_1 and A_2 with only one operation, the coefficients A_1 and A_2 are adjusted in increments. Therefore, multipliers 226 and 228 multiply the corresponding outputs of the correlators 214 and 216 by a quantity α, which has a value of less that one. The output of the multiplier 226 is added to coefficients A_1, and the output of the multiplier 228 is added to the coefficients A_2. The value α is used so that the coefficients A_1 and A_2 are corrected in small increments in order to ensure a smooth convergence.

A third embodiment (not shown) of the converger 7 may be provided for the dual path equalizer 60. In this case, the adaptive coefficient control according to the third embodiment would have the same general arrangement as the adaptive coefficient control 210 shown in Figure 15. However, the inverse first and second post-processors in this third embodiment would be different from the inverse first and second post-processors 222 and 224 of the adaptive coefficient control 210 because of the difference between the first and second post-processors 48 and 54 of the dual path equalizer 40 and the first and second post-processors 68 and 74 of the dual path equalizer 60.

A fourth embodiment of the converger 7, i.e., an adaptive coefficient control 240, may be provided for the triple path equalizer 80 and is shown in Figure 16. The adaptive convergence control 240 includes a conjugater 242 which conjugates the data from the 2xFFT 84. The output of the conjugater 242 is supplied to first, second, and third correlators 244, 246, and 248. An error generator 250 generates an error based upon the output of the triple path equalizer 80. The adaptive convergence control 240 includes an up sampler 252 which up samples by two the output of the

error generator 250 in order to thereby reverse the effects of the down sampler 106. The adaptive coefficient control 240 also includes an inverse first post-processor 254, an inverse second post-processor 256, and an inverse third

5   post-processor 258. The inverse first, second, and third post-processors 254, 256, and 258 produce an inverse of the first, second, and third post-processors 88, 94, and 100 of the triple path equalizer 80. The outputs of the inverse first, second, and third post-processors 254, 256, and 258

10   are supplied to the corresponding correlators 244, 246, and 248.

The correlators 244, 246, and 248 multiply (i) the up sampled error from the error generator 250 as processed by the respective inverse first, second, and third post-

15   processors 254, 256, and 258 and (ii) the conjugated output of the 2xFFT 84. In effect, the outputs of the correlators 244, 246, and 248 are adjustment values of the coefficients A_1, A_2, and A_3 that, when added to the existing coefficients A_1, A_2, and A_3 applied by the triple path equalizer 80,

20   would cause the triple path equalizer 80 to substantially eliminate ghosts from the received signal.

Multipliers 260, 262, and 264 multiply the outputs

of the correlators 244, 246, and 248 by a quantity α, which

has a value of less that one. The output of the multiplier

260 is added to coefficients A_1, the output of the

5   multiplier 262 is added to the coefficients A_2, and the

output of the multiplier 264 is added to the coefficients

A_3.

A fifth embodiment of the converger 7, i.e., an

adaptive coefficient control 270, may be provided for the

10   triple path equalizer 110 and is shown in Figure 17. The

adaptive convergence control 270 includes a conjugater 272

which conjugates the data from the 2xFFT 114. The output of

the conjugater 272 is processed along first, second, and

third control paths 274, 276, and 278 which correspond to

15   the first, second, and third paths 122, 128, and 136 of the

triple path equalizer 110. The first control path 274

includes a left one-sample shifter 280, a first by-two down

sampler 282, and a first correlator 284. The left one-

sample shifter 280 and the first by-two down sampler 282

20   replicate the processing of the left one-sample shifter 116

and the first by-two down sampler 118 in the first path 122

of the triple path equalizer 110. The second control path

- 46 -

276 includes a second by-two down sampler 286 and a second

correlator 288. The second by-two down sampler 286

replicates the processing of the second by-two down sampler

124 in the second path 128 of the triple path equalizer 110.

5   The third control path 278 includes a right one-sample

shifter 290, a third by-two down sampler 292, and a third

correlator 294. The right one-sample shifter 290 and the

third by-two down sampler 292 replicate the processing of

the right one-sample shifter 130 and the third by-two down

10   sampler 132 in the third path 136 of the triple path

equalizer 110. An error generator 296 generates an error

based upon the output of the triple path equalizer 110.

The error from the error generator 296 is supplied

to the correlators 284, 288, and 294. The correlators 284,

15   288, and 294 multiply the error from the error generator 296

and the respective outputs of the first, second, and third

by-two down samplers 282, 286, and 292. In effect, the

outputs of the correlators 284, 288, and 294 are adjustment

values of the coefficients A_1, A_2, and A_3 that, when added

20   to the existing coefficients A_1, A_2, and A_3 applied by the

triple equalizer 110, would cause the triple path equalizer

110 to eliminate ghosts from the received signal.

Multipliers 298, 300, and 302 multiply the outputs of the correlators 284, 288, and 294 by a quantity α, which has a value of less that one. The output of the multiplier 298 is added to the existing coefficients A_1, the output of

5   the multiplier 300 is added to the existing coefficients A_2, and the output of the multiplier 302 is added to the existing coefficients A_3.

By the same token, the adaptive coefficient control 270 can be generalized in order to converge the

10   output of the multipath equalizer 140 into a ghost free signal, and the adaptive coefficient control 210 can be generalized in order to converge the output of the multipath equalizer 160 into a ghost free signal.

An error generator 400 is shown in Figure 18 and may be used for the error generators 188, 218, 250, and 296.
The adaptive coefficient controls of the present invention may be operated in either a training mode or a data directed mode. During the training mode, a training signal is transmitted by a transmitter to the corresponding equalizer.

20   A summer 402 subtracts the training signal (which may be locally stored or generated) from the output of the equalizer (which is the received training signal) and

supplies this difference to a switch 404. During the data

directed mode, the data output of the equalizer is sliced by

a slicer 406 using stored slice levels 408 in order to

generate as an output the one stored slice level which is

5   closest to the received data. The stored slice levels 408

are the reference in the data directed mode. A summer 410

subtracts the stored slice level provided by the slicer 406

from the output of the equalizer and supplies this

difference to the switch 404. The switch 404 selects either

10   the output of the summer 402 or the output from the summer

410 as the error which is processed by the particular

adaptive coefficient control according to the various

descriptions above. The switch 404 may be operated by the

synchronizer 6 shown in Figure 2.

The switch 404 may select the output of the summer

402 (i.e., the training mode) during system start up when

the A coefficients do not have values likely to produce

meaningful output data. Once the equalizer has been

coarsely converged during the training mode, the switch 404

20   may be switched to select the output of the summer 410

(i.e., the data directed mode) so that data can be used in

order to achieve and maintain precise convergence of the equalizer.

Certain modifications and alternatives of the present invention have been discussed above. Other modifications and alternatives will occur to those practicing in the art of the present invention. For example, because the present invention operates most satisfactorily in the presence of ghosts and other linear distortions, the term ghost as used herein in connection with the present invention includes ghosts and/or other linear distortions.

Also, the Fast Fourier Transforms and inverse Fast Fourier Transforms described above can have lengths other than those described above.

Furthermore, the invention has been described above as if a single ghost is received. In the case where multiple ghosts are received, it may be desirable to apply multiple sets of the coefficients to the received signal or a single set tailored for multiple ghosts. Also, the spacing between coefficients is described above as being the interval d. However, in the case where the interval d is not evenly divisible into the block length of a data block, or in the case where more than one ghost are received, the

spacing between the coefficients may be other than the interval d.

The guard interval discussed above may have any desired values including zero.

Moreover, as described above, the error generated by the error generator 400 is correlated with the conjugated output of a 2xFFT that is part of the corresponding equalizer, and this correlation is used to adjust the appropriate **A** coefficients. Instead, the error generated by the error generator 400 could be used in a zero forcing method to directly adjust the appropriate **A** coefficients without correlation with the conjugated output of the corresponding 2xFFT. The zero forcing method, however, may not suitably account for noise in the input.

Accordingly, the description of the present invention is to be construed as illustrative only and is for the purpose of teaching those skilled in the art the best mode of carrying out the invention. The details may be varied substantially without departing from the spirit of the invention, and the exclusive use of all modifications which are within the scope of the appended claims is reserved.